

# Towards Privacy-Aware Handling of Authorizations

Wolfgang Dobmeier, Günther Pernul  
Department of Information Systems  
University of Regensburg  
93040 Regensburg, Germany

{ wolfgang.dobmeier, guenther.pernul }@wiwi.uni-regensburg.de

## Abstract

*Privacy issues have hindered centralised authentication approaches from being adopted by a wide range of users. This also applies to authorizations which suffer from privacy problems when stored and processed centrally. We present first steps towards a framework of privacy-aware handling of authorizations. We split up the storage and the processing of access control policies in a user-centric approach. We illustrate our approach at the example of a security infrastructure scenario.*

## 1. Introduction and Motivation

With the still growing number of users and resources having access to and being contained in IT systems, the need for prevention of unauthorized use of resources is evident. This is commonly accomplished via access control mechanisms that implement a specific access control model. A well-known implementation of access control is the concept of reference monitor, which centrally intercepts access requests and denies or allows them. The behaviour of this mechanism is governed by an according policy which contains rules defining the access rights of users on resources. We denote the granting of a permission using the term authorization [11]. In a technical sense this comprises two aspects: first, an entry into an access control policy allowing an operation of a user on a resource, i.e. of a subject on an object; and second, the process of deciding if a user has the permission to execute a specific operation on a specific resource at the time of the access request and the subsequent enforcement of the decision.

Another key component of today's IT-security requirements is privacy, which strives to protect personally identifiable information (PII) of individuals. To this end, several approaches have evolved, e.g., policy languages for specification of privacy preferences, techniques for attaching privacy policies to PII and ensuring their enforcement, and

methods for privacy-respecting data mining.

Security functionality like authentication and access control is influenced by privacy issues, too. This becomes evident when we look at the history of the Microsoft .NET Passport initiative as an example. This infrastructure provided a single sign-on service for websites. User account data consisting of user name and password were kept by a central entity under control of one organization, which authenticated the users trying to sign on. Obviously, the approach failed. A main reason for this was that the users did not have enough trust in one single organization properly processing handling the account data. This raised huge privacy concerns [11] because Microsoft could see each user authentication taking place at one of the websites. To overcome these concerns and limitations, later single sign-on approaches like the Liberty Alliance Project [17] have distributed the user's identity information to several entities in the infrastructure.

In this context, an area that so far has been paid little attention to is the privacy-aware handling of authorizations. As access control policies are so far stored and processed centrally in many architectures of distributed systems, problems arise as one policy repository and thus the central access control implementation knows all the authorizations any subject that is registered possesses. Besides possible availability and performance issues, the central security module can monitor all authorization requests (analogously to the Passport initiative), so profiles of users' behaviour could be created. Consequently, although these central security components are believed to be trusted, it is desirable from a privacy point of view that no unnecessary information should be processed there and the user should have some degree of control on the processing and storage of his authorizations.

Hence, in this paper we propose to partition and subsequently distribute access control policies and their processing to several distinct entities in the security infrastructure. This is done in order to protect the privacy of users from misbehaviour of central security components.

Our approach solves a part of the privacy problem arising from central security service components in infrastructures, as they have been described e.g. in [7]. We focus on identity-based access control models in this paper, as described in Section 2.

In addition to the term Authorization introduced above, we use the term Policy Decision Point (PDP) as defined in the XACML standard [12]. A PDP receives an access control decision request and retrieves the applicable policies. Then the access control decision is computed and sent back to the requesting entity (i.e., the entity enforcing the decision).

This work is partly performed within the European IST project Access-eGov<sup>1</sup> focusing on the development of a semantic eGovernment infrastructure supporting citizens' security and privacy needs to a high degree.

The paper is organized as follows. We give an outline of our partitioning approach in Section 2. Section 3 describes a sample application scenario. Section 4 presents related work; conclusions are given in Section 5.

## 2. Partitioning Authorizations

We investigate two different aspects in our proposal. The first one is how access control policies can be split into smaller pieces that are to be stored in a distributed manner; the second one is the processing of authorizations at the time of the access control decision. This is in line with the twofold nature of authorizations as described in Section 1.

The following work can be seen as steps towards a framework from which concrete implementations of privacy-aware processing of authorizations can choose and combine different aspects to satisfy defined privacy needs. We also follow the paradigm of user centricity which in our context means to give the user at least some degree of control over transactions involving his access control data. This is in analogy to user centricity in identity management infrastructures where identity data is processed [3].

### 2.1. Segmentation and Storage of Authorizations

In our model, a policy  $P$  consists of a set of authorizations. Formally, we define  $P = \{A_1, \dots, A_n\}$ . The total sets of subjects  $S$  and objects  $O$  in the system are represented by the sets  $S = \{S_1, \dots, S_m\}$  and  $O = \{O_1, \dots, O_k\}$  consisting of individual subjects  $S_i$  requesting access to objects in  $O$ . There exist different operations  $Op_i$  users can execute on resources. An authorization  $A = (S_i, O_j, Op)$  grants subject  $S_i$  the right to execute one or more operations on object  $O_j$ , which are defined through the set  $Op$ .

<sup>1</sup><http://www.access-egov.org/>

	1	2	3	...
A	r	r,w	r	..
B	w	r,w,d	-	..
C	w	w	u	..
...	..	..	..	..

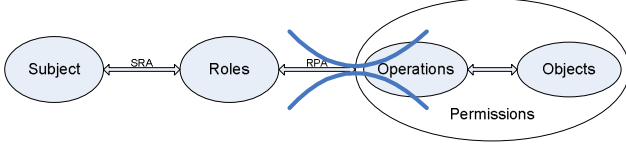
Figure 1. Partitioning Access Matrices

First we develop some criteria on how authorizations can be segmented. We can group these criteria into three main approaches. First, policies and the processing can be split in a subject-oriented way, i.e. according to users' identities or roles. Second, object-oriented segmentation like task-based or service-based partitioning can take place. Third, a variant consisting of random splitting and random distributed processing of policies is possible.

Now we describe how authorizations in traditional identity-based access control can be split up. Authorizations in traditional access control models [14] are based on the identity of subjects and objects. Users as well as resources are known a-priori. In this first step towards privacy-aware handling of authorizations, we stick with the basic forms of access-matrix based models and role-based models. Authorizations in other traditional models, i.e. the family of multilevel models like the well-known military model, can not be distributed because dedicated authorizations do not exist. Instead, mandatory system-wide authorization rules are in effect (e.g., the combination of a no-read-up and a no-write-down rule) and privileges are determined by evaluating metadata of subjects and objects in the form of security classes. In attribute-based access control models (e.g. [4]), which base authorizations on the attributes of subjects and objects, the determination of a specific user's authorizations usually is not possible because not all the attributes a user possesses may be known beforehand to the security infrastructure. Moreover, suitable approaches for hiding policies, attributes, and even actual values of attributes have recently been developed in [9, 10, 15].

In access matrix-based models, authorizations directly link subjects to objects. Each allowed set of operations of a subject on an object leads to an entry in the access matrix as depicted in Figure 1, with subjects corresponding to rows and objects corresponding to columns. We define an access matrix as the mapping  $M : S \times O \rightarrow Op$ . An entry  $m_{i,j}$  holds the set of permitted operations subject  $S_i$  can execute on object  $O_j$ . There are three popular implementations of the access matrix:

- Authorization tables, which hold authorizations  $A_i$  for non-empty entries of  $M$ ,
- Capability lists, which hold tuples  $(O_j, Op)$  for a specific  $S_i$ ,



**Figure 2. Partitioning RBAC Authorizations**

- Access control lists (ACLs), which hold tuples  $(S_i, Op)$  for a specific  $O_j$ .

These implementations can easily be transferred into our representation. Figure 1 shows an exemplary split of an access matrix. Given a matrix  $M$ , we divide it into  $n$  disjunct submatrices by defining mappings  $M_i : s \times o \rightarrow Op$  with  $s \in S, o \in O, i = 1, \dots, n$  and  $(s, o)$  being different for each  $M_i$ . The  $M_i$  represent the partitions of the original policy that can be stored and processed independently.

Role-based Access Control (RBAC) [8] assigns privileges, i.e. tuples of  $(O_j, Op)$ , to roles and furthermore users to roles. Thus, authorizations in RBAC do not directly link subjects to objects; instead, roles are used as intermediaries. RBAC is session-based, which means that users can select different roles for activation during a session. This is done in order to support the least-privilege principle and requires that all roles a user can potentially activate are known at the time of opening the session. Advanced forms of RBAC introduce role-hierarchies and constrain the way roles can be assigned to users; we do not consider these topics here. Formally, we have a set of roles  $R$  and relations  $SRA \subseteq S \times R$  representing the subject-role-assignment and  $RPA \subseteq R \times (O_j, Op)$  for the role-privilege-assignment. We propose to split up RBAC authorizations as shown in Figure 2, i.e. we segment the  $RPA$  relation into disjunct partitions consisting of sets of tuples  $(r, O_j, Op), r \in R$  to be stored at separate entities. A split at the  $SRA$  relation would require the security infrastructure to time-consumingly gather a user's potential roles from all over the infrastructure. Otherwise, the session-based least privilege principle could not be implemented.

We now briefly discuss issues arising from the distributed storage of the partitions, i.e. the submatrices and the segments of the role-permission-assignment relation. The segments have to be non-overlapping and non-conflicting, as resulting from the splitting method introduced above. Following the user-centricity paradigm, the user is in control where his partitions are stored and which PDPs have access to each of the partitions. The main design choice is whether the policy segments are pushed to or pulled by the PDP that processes them. In the pull case, the question arises how the partitions can be located by the PDP and retrieved from the infrastructure. There could be a central index to the different partitions each PDP can ac-

	Centralised Storage	Distributed Storage
Centralised PDP	All authorizations and their usage are known to a single entity.	Each authorization process, but only part of a user's authorizations is known to a specific PDP.
Distributed PDPs	All authorizations are known to a single entity but not the time of their usage.	Knowledge on user's potential and performed authorizations is distributed among distinct entities.

**Table 1. Privacy implications of policy storage and evaluation**

cess, or the PDP could search every possible storage location in the infrastructure for appropriate policy segments. The first option implicates the need for another central entity in the infrastructure, while the second approach is less performant. As for the push paradigm, the client could explicitly state the relevant partition's location when issuing requests. Similarly, the user could retrieve the relevant partitions by herself and send them to the PDP, which results in an Akenti-like approach [16] where users can hold certificates containing access control policies.

## 2.2. Controlling the Processing of Authorizations

Access control policies in distributed systems can be processed in various ways. We can differentiate these ways with respect to where the policy is stored and where the PDPs are located. Either can be done in a centralized or distributed manner, which boils down to four possible combinations of policy processing; the privacy implications of these design choices are shown in Table 1. To reach a desired level of privacy, a security infrastructure should provide the corresponding functionality.

Besides the partitioning approach presented in section 2.1, the user should be given control by the security infrastructure as to where his authorizations are evaluated. This should be provided as a base feature of the infrastructure, because distributed evaluation of policies can be applied to all kinds of access control models. This prevents single PDPs from obtaining complete user access request profiles also for attribute-based and multilevel models. In case of the distributed processing of identity-based models, the user has to authenticate herself at each PDP which can be ensured by the use of assertions from a trusted identity provider, e.g. in the form of SAML [13] assertions.

### 3. An Application Scenario

We present a sample eBusiness-scenario to show the applicability of the concepts introduced above.

Two financial institutions (Bank A and Bank B) and a central identity provider as well as two PDPs with accompanying policy repositories are members of a security infrastructure. PDP A is operated by Bank A, PDP B is maintained by the security infrastructure provider. Bank B is not interested in maintaining any security services and instead relies on the security infrastructure and accepts decisions from both PDPs. User Alice is an employee of Bank A; her authorizations for her daily work at Bank A are stored in the policy repository of Bank A's PDP and also evaluated there.

As Alice is in need of money, she decides to raise a credit at Bank B because she does not want her employer (Bank A) to know about it. To this end, she first needs to authorize Bank B to read her personal identity information which is stored at the identity provider of the security infrastructure. Alice chooses to separate this authorization and stores it in the repository of PDP B maintained by the infrastructure. She does so using the functionality provided by her digital personal assistant. At last, Alice proceeds to close the loan contract and has the authorization for the reading of her personal data processed at PDP B. Hence, Bank A did not learn about Alice's loan.

### 4. Related Work

There exists a large body of work on traditional distributed authorization systems. Lopez et al. [11] provide a comprehensive survey. There are different approaches to handling authorizations in these systems. No one explicitly addresses privacy issues. While there are systems that enable distributed storage of PII (e.g. [17]) and authorizations (e.g. [16, 5, 1]), users cannot control the processing of authorizations in these proposals. Chadwick et al. [6] present an approach for stateful coordination of distributed PDPs which is not privacy-aware.

Another important area is privacy-enhanced access control. There have been approaches for purpose-based and obligation-aware policies and languages, especially in the context of the PRIME project [2]. Data handling policies are used to govern the release of PII to third parties and their further processing at the receiving site; sanitized policies are employed to protect sensitive information about policies during credential-based access control negotiation. Our approach is orthogonal to these proposals in that we address a new aspect of privacy-enhanced access control.

### 5. Conclusions

We presented basic concepts towards a privacy-aware handling of authorizations in policy-governed systems. The approach was described with focus on identity-based access control models. We see the benefit of the approach not only for users of security infrastructures but also for providers. This is because of an increased demand for protection of users' PII in the digital economy; thus, increased privacy poses a competitive advantage for providers of security infrastructures. The concepts in this paper outline first steps towards this goal.

On the user side, the acceptability of our approach will depend on the availability of easy-to-use tools for managing storage and processing of authorizations. We are planning to evaluate our approach in the Access-eGov eGovernment project. In this project, a personal digital assistant will be developed as an interface to the security infrastructure. This assistant will support user-friendly management of privacy preferences in general; this also includes functionality for privacy-aware handling of authorizations. Furthermore, we will examine the splitting of authorizations in advanced access control models like e.g. extended forms of RBAC and provide a XACML-based implementation of our approach. We will also investigate privacy-aware generation and storage of audit data in security infrastructures.

**Acknowledgements.** A part of this work has been carried out within the European Project Access-eGov (contract no. FP6-2004-27020). The views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect the views of the European Commission.

### References

- [1] R. Alfieri, R. Cecchini, V. Ciaschini, L. dell'Agnello, Á. Frohner, A. Gianoli, K. Lörentey, and F. Spataro. Voms, an authorization system for virtual organizations. In F. F. Rivera, M. Bubak, A. G. Tato, and R. Doallo, editors, *European Across Grids Conference*, volume 2970 of *Lecture Notes in Computer Science*, pages 33–40. Springer, 2003.
- [2] C. A. Ardagna, E. Damiani, S. D. C. di Vimercati, and P. Samarati. Towards privacy-enhanced authorization policies and languages. In *Proc. of the 19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec 2005)*, pages 16–27, 2005.
- [3] A. Bhargav-Spantzel, J. Camenisch, T. Gross, and D. Sommer. User centrality: A taxonomy and open issues. In *Proc. of the Second ACM Workshop on Digital Identity Management (DIM 2006)*, pages 1–10, New York, NY, USA, 2006. ACM Press.
- [4] P. Bonatti and P. Samarati. A unified framework for regulating access and information release on the web. *Journal of Computer Security*, 10(3):241–272, 2002.

- [5] D. W. Chadwick and A. Otenko. The permis x.509 role based privilege management infrastructure. *Future Generation Comp. Syst.*, 19(2):277–289, 2003.
- [6] D. W. Chadwick, L. Su, A. Otenko, and R. Laborde. Coordination between distributed pdps. In *Proc. of the Seventh IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY '06)*, pages 163–172, 2006.
- [7] F. Dridi, M. Fischer, and G. Pernul. Csap - an adaptable security module for the e-government system webocrat. In *Proc. of the IFIP TC11 18th International Conference on Information Security (SEC 2003)*, pages 301–312, 2003.
- [8] D. F. Ferraiolo, R. S. Sandhu, S. I. Gavrilu, D. R. Kuhn, and R. Chandramouli. Proposed nist standard for role-based access control. *ACM Transactions on Information and System Security*, 4(3):224–274, 2001.
- [9] K. B. Frikken, M. J. Atallah, and J. Li. Attribute-based access control with hidden policies and hidden credentials. *IEEE Transactions on Computers*, 55(10):1259–1270, 2006.
- [10] J. Li and N. Li. Oacerts: Oblivious attribute certificates. *IEEE Transactions on Dependable and Secure Computing*, 3(4):340–352, 2006.
- [11] J. Lopez, R. Oppliger, and G. Pernul. Authentication and authorization infrastructures (aais): A comparative survey. *Computers & Security*, 23(7):578–590, 2004.
- [12] OASIS. extensible access control markup language (xacml) version 2.0, 2005.
- [13] OASIS. Security assertion markup language (saml) version 2.0, 2005.
- [14] P. Samarati and S. D. C. di Vimercati. Access control: Policies, models, and mechanisms. In R. Focardi and R. Gorrieri, editors, *Foundations of Security Analysis and Design*, volume 2171 of *Lecture Notes in Computer Science*, pages 137–196. Springer, 2000.
- [15] A. C. Squicciarini, E. Bertino, E. Ferrari, and I. Ray. Achieving privacy in trust negotiations with an ontology-based approach. *IEEE Transactions on Dependable and Secure Computing*, 3(1):13–30, 2006.
- [16] M. R. Thompson, A. Essiari, and S. Mudumbai. Certificate-based authorization policy in a pki environment. *ACM Transactions on Information and System Security*, 6(4):566–588, 2003.
- [17] T. Wason. Liberty id-ff architecture overview version 1.2. Liberty Alliance Project, 2006.